COLARDYN IT GCV

HWBOT Data API

Prepared for: 3rd party benchmark application writers
Prepared by: Frederik Colardyn & Dennis Devriendt, HWBOT developers

27 Dec 2012
Compatible with HWBOT Version: 5.0.0

Version 1.1

COLARDYN IT GCV

Table of Contents

COLARDYN IT GCV

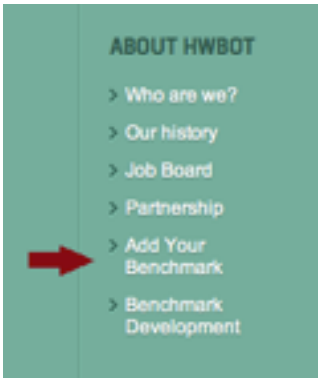
Adding your benchmark

Objective

We want to make it easier for third parties to add their benchmark application to HWBOT. Adding a new benchmark to the site can now be done using a new web interface. In addition to this, we have a new API which will make it a lot easier to integrate HWBOT in your application.

Solution

To register your application, click on the “Add Your Benchmark” link on the bottom of http://hwbot.org . This will start the wizard. To access it, you need a HWBOT account and you need to be logged in. As soon as you’ve add your benchmark to HWBOT, your account will become the sole administrator for that benchmark. To edit the settings for your application later on, go to your user profile (http://hwbot.org/profile). A list of benchmarks that you registered with your account should appear on the bottom, under “Other Settings”.



General Settings

FIrst, you have to provide the name of your benchmark and a download link. This doesn’t have to be a direct link, we just provide the link to the users so that they can easily download your application. If you choose to do so, you can also provide a link to your website homepage. Note that before continuing to the next step, you have to accept the usage agreement.

Score Settings

In the next step, we want to know a bit more about your benchmark scoring mechanism.

- **score units**: If your benchmark has a specific score unit, you can add it here. Score units cannot be empty though. If your application doesn’t have a specific score unit, just use “points” or something similar. If the score is a time measurement, use “seconds”. That way, we can show the time in a nice format on HWBOT.
- **hardware component**: On HWBOT, we group benchmarks according to the hardware component that they stress. Most benchmarks mainly stress the processor or the videocard if it’s a 3D benchmark. Currently, you can only choose one.
- **names of subtests:** If your benchmark consists of multiple subtests, it is possible to add the score for each subtest to a submission. You can provide names for each subtest here. An example is provided in the Submission API documentation.
- **multithreaded:** check this if your benchmark scales well with multiple cpu cores/ multiple graphics cards. If this is checked, HWBOT keeps track of seperate rankings for each number of cpu cores / graphics cards.

COLARDYN IT GCV

- **users are allowed to manually submit scores:** HWBOT has been working on a API that benchmarks can use to automatically submit results to HWBOT, so users don't have to. For more information, please read the next section "Submission API". If you choose to not use this API, this option has to be checked so that users can manually add their results.

Security Settings

These settings only apply if your benchmark will use the Submission API. The API requires an xml datafile that contains the score and hardware information. To prevent cheating, the datafiles that your benchmark will submit should be encrypted. Another option is to use a checksum to prove authenticity but this is less secure as users could easily guess the checksum algorithm and just generate the checksum for their tampered datafiles. Plaintext datafiles without checksums should only be used for testing. If the datafiles are rather big, you could compress them using GZIP. If you do so, the compressed datafiles option has to be checked. How to use these security options is explained in-depth in the next section, "Submission API".

Encryption algorithms supported: AES and DESede (also known as 3DES) in CBC and ECB mode.

Hashing algorithms supported: SHA1, SHA256 and MD5.

Other Settings

If all went well, your benchmark should now be registered and you should be taken to a little get started with the Submission API guide. If you want to review or change settings, follow the link on the bottom of your user profile page. This page contains all the settings that you've entered using the wizard, with the addition of two more, version and status.

- **version:** If you release a new version for your benchmark, and scores are not comparable with previous versions, you will have to prevent users from using the old version. The Submission API does a version check. If a datafile has a version that is not included in the allowed versions field, the submission will be rejected.

- **status:** When you register your benchmark, it will have the alpha status and will still be hidden from users on HWBOT. This is useful when you are still in the process of testing the Submission API. Set the status to beta if testing is done, the benchmark will then be made visible on our site. There is a third status, released, which means users can receive HWBOT points for their submissions. Only HWBOT can set the status to released.

COLARDYN IT GCV

Submission API

Objective

The HWBOT Submission API allows third party applications to submit a benchmark score in a secure way, removing the need for users to submit their benchmark score by manually completing a web form.

The goal is not to bypass the HWBOT web interface completely, but to provide a less error prone and more user friendly way to share and compare benchmark scores . It may be that the user has to complete the submission by providing details the third party benchmark was not able to give, for example the cooling that was used. After a successful submission, the user can view and compare their benchmark score on HWBOT.

Goals

- security: ensure the benchmark data is genuine
- usability: remove the need for users to enter benchmark data manually

Solution

The HWBOT Submission API is a HTTP REST interface. Data is sent as a XML file, using a HTTP POST request to the HWBOT server. The format of this XML file is explained further on. To provide security, encryption and a checksum mechanism can be used.

Sending Data

URL: http://hwbot.org/submit/api

Request parameters:

- **client**: text, required. The name of your benchmark. If the name contains spaces, replace them with underscores.
- **clientVersion**: text, optional. This defines the version of the benchmark. E.g. “1.0.3”. The suggested format is <major>.<minor>.<patch level>, but is not enforced. This field allows changing the behavior in future releases of either the third party application or the HWBOT engine, and remain compatible.
- **always200OK**: boolean, optional, default false. When “true”, the response will always be 200 OK, regardless whether posting the data was a success or not. This is required for some frameworks.
- **strict:** boolean, optional, default false. When “true”, the xml contained in the datafile will be validated against our XSD Schema. If the xml file file is not valid, an error will be returned. This is useful for testing purposes.

- **checksum**: text, required if you enabled checksum validation in the benchmark security settings. This is a hexadecimal representation of the checksum (hash) of the data file.

- **data**: file, required. This file contains the submission info. It can be a UTF-8 plaintext XML but we recommend encrypting the file to prevent tampering. The data can be submitted as a file attachment of a POSTed form OR as body content of the request, e.g. for C#: WebRequest.uploadFile() method.

Response

The response can either be formatted in **JSON or in plain XML**, depending on the request header. Either way, the **response** object contains the following fields:

- **status**: success or error
- **url**: in case of success, this is the url were the user can complete his submission. The client is required to open this url in the default browser of the user. If this does not occur, the submission data will be lost after 30 minutes. In case of error, this field is optional. If it's included, it is the url where more information can be found about the error.
- **message**: in case of success, this field will contain "Submission successfully received.". In case of error, this field contains a user friendly error message.
- **technicalMessage**: in case of success, this field will be empty. In case of error, this will contain a technical reason why there was a failure. The end user should not see this error, it is for debugging purpose only.

The response HTTP codes are:

- **200 OK**: success
- **400 BAD REQUEST**: required parameter missing, outdated client version, failed to decrypt or parse the data file, etc.
- **500 INTERNAL SERVER ERROR**: uncaught error on our part, shouldn't happen

Example code

To test if HWBOT accepts a datafile, you can use this simple HTML code example:

```
<form action="http://hwbot.org/submit/api" enctype="multipart/form-data" method="post">
        <input type="file" name="data" />
        <input type="text" name="client" value="My Benchmark" />
        <input type="text" name="clientVersion" value="1.0.0" />
        <input type="text" name="checksum" value="1234567890ABCDEF" />
        <input value="submit" type="submit" />
</form>
```

example in Java:

```
DefaultHttpClient httpclient = new DefaultHttpClient();
try {
    HttpPost post = new HttpPost("http://hwbot.org/submit/api/");
    MultipartEntity reqEntity = new MultipartEntity();
    reqEntity.addPart("data", new FileBody(new File("datafile.hwbot")));
    reqEntity.addPart("client", new StringBody("benchmark_name"));
    // HTTPClient throws an exception if the HTTP status isn't 200 OK
    reqEntity.addPart("always200OK", new StringBody("true"));
    reqEntity.addPart("strict", new StringBody("true"));
    post.setEntity(reqEntity);

    System.out.println(httpclient.execute(post, new BasicResponseHandler()));
```


```
} catch(Exception e) {
    System.err.println(e);
} finally {
    httpclient.getConnectionManager().shutdown();
}
```

Datafile security

When the submission API receives a data file, processing is done in 4 steps.

1. If the application provides a checksum, we first calculate the hash of the file and compare it to the given checksum.
2. If it matches, and the application encrypts datafiles, decrypt the datafile using the correct cipher.
3. If the application also GZIPs the datafile, decompress it.
4. The result is a plaintext UTF-8 XML file.

We encourage third parties to encrypt the datafiles. A checksum provides some security, but malicious users can fairly easy construct their own submission file and create the checksum using the correct hashing algorithm. Therefore, if encryption is enabled, the checksum also **must be** encrypted!

To help third-parties understand the Submission API, we created 4 different files:

HWBOTSubmitSchema.xsd http://static.hwbot.org/xsd/HWBOTSubmitSchema.xsd : XML Schema for a submission file.

GenericComplete.xml http://static.hwbot.org/xsd/GenericComplete.xml : Example plaintext submission file

GenericCompleteNotGZipped.hwbot http://static.hwbot.org/xsd/GenericCompleteNotGZipped.hwbot : same file, but encrypted (key: DE898DB7F16753C8FA550B1089937504, IV: 1234567890ABCDEF1234567890ABCDEF)

GenericCompleteGZipped.hwbot http://static.hwbot.org/xsd/GenericCompleteGZipped.hwbot : same file but compressed before encryption.

example with the provided xml file GenericComplete.xml, using AES-128-CBC encryption, MD5 hash and no compression.

encryption key: DE898DB7F16753C8FA550B1089937504, encryption IV: 1234567890ABCDEF1234567890ABCDEF:

- `openssl aes-128-cbc -K DE898DB7F16753C8FA550B1089937504 -iv 1234567890ABCDEF1234567890ABCDEF -in GenericComplete.xml -out GenericCompleteNotZipped.hwbot`

now determine the MD5 hash (in hexadecimal representation: 9b607589be525aa30de99c4e44e3d523):

- `openssl dgst -md5 -binary GenericCompleteNotZipped.hwbot > hash`

finally, encrypt the hash the same way the datafile was encrypted (make sure that you encrypt the bytes, not the hexadecimal characters!). For our example, the result is:

- AFE908CCA771FDE34165C924BB5BA48CB24A8187B7D287D209831BFD77320413, this is the checksum that should be provided to the Submission API.

The encryption key and iv can be chosen freely on the application administration site, but make sure that malicious users can't easily find the key.

Datafile xml format

The file HWBOTSubmitSchema.xsd is a schema for the datafile format that our Submission API requires. The GenericComplete.xml is an example submission file that uses most elements. We'll give a short explanation for each element. We don't check if a datafile validates with the schema, it is just provided as a guideline. Applications aren't required to provide this much information, but the more information, the better of course!

Note that number values have to have a point '.' as a decimal mark. Temperature values are in degrees Celsius.

```
<?xml version="1.0" encoding="utf-8"?>
<submission xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://hwbot.org/submit/api">
  <application>
    <name>required: Exact name of the application</name>
    <version>required: Version number of the application</version>
  </application>
  <score>
    <points>required: total points scored</points>
    <subscores>
      <subscore name=”name of the subtest”>list of subscores (for example for each test)</
subscore>
    </subscores>
  </score>
  <timestamp>timestamp of when the benchmark was run, might be useful to detect tampering if your
application only uses the Submission API for submitting scores</timestamp>
  <screenshot contentType="image/jpg" >Base64 encoded verification image, can be a jpg or png. If
contentType isn’t specified, the image will be  converted and saved as a jpeg on our server.
  </screenshot>
  <verificationUrl>if your application has implemented verification urls, you can enter it here</
verificationUrl>
  <applicationChecksum>If your application creates checksums in any way, it can be entered here.
If a checksum is provided, we show ‘valid’ on the submission page for this submission. If you
want to use the mechanism as described in the previous section, the checksum has to be provided
as a request parameter.</applicationChecksum>
  <hardware>
    <processor>
      <name>cpu model</name>
      <amount>physical processors on the motherboard</amount>
      <idleTemp>temperature before the benchmark started</idleTemp>
      <loadTemp>temperature when the processor is under load</loadTemp>
      <coreClock>cpu frequency</coreClock>
    </processor>
    <videocard>
      <name>gpu model</name>
      <vendor>gpu vendor</vendor>
      <amount>physical graphics cards</amount>
      ... etc, same settings as for the processor can be provided for other hardware components.
See the schema for the full definition and GenericComplete.xml for an example that uses most
tags.
    </videocard>
    <motherboard>
      <name>motherboard model</name>
      <vendor>motherboard vendor</vendor>
      <chipset>chipset used, if the motherboard model is provided, chipset and vendor are auto-
matically filled in on HWBOT.</chipset>
```

```
    </motherboard>
    <memory>
      <vendor>memory vendor</vendor>
      <type>DDR3 SDRAM, DDR2 SDRAM, etc. Use the exact naming as on hwbot.org</type>
      <rating>e.g. PC8500, use the exact naming as on hwbot.org</rating>
      <totalSize>total size of all the installed DIMMs</totalSize>
      ... timing tags can follow here.
    </memory>
    <disk>
      <vendor>vendor</vendor>
      <amount>number of disks</amount>
      <diskCapInGB>capacity per disk, in gigabytes</diskCapInGB>
      <series>if your application nows the exact series of the disk, it can be entered
here.</series>
      <configuration>again, if your application knows the configuration, for example RAID-5, it
can be provided in this xml file.</configuration>
    </disk>
  </hardware>
  <software>
    <os>
      <family>OS family: 'Windows', 'Linux' or 'Mac and others', currently not shown on HWBOT
submission pages, but we might in the future.</family>
      <fullName>OS full name, for example Microsoft Windows 7 Professional</fullName>
    </os>
  </software>
  <metadata name="metadata_name">
Here, your application can include a log file or configuration file. It will not be visible on
the submission page for regular users, but moderators can view it and use it to find possible
malicious use.
  </metadata>
</submission>
```

COLARDYN IT GCV

Version Check API

Objective

The HWBOT Version Check API allows third party applications to check which is the latest supported version by HWBOT. Using the Submission API with a different version might be rejected by HWBOT, depending on the strategy chosen when this third party application was first added to the HWBOT suite. The default approach is to reject any version not matching the exact version string.

Goals

- usability: inform the user up front whether a newer version of the benchmark application is required to make a submission to HWBOT

Solution

The HWBOT Submission API is a HTTP REST interface. Data is sent using a HTTP GET request to the HWBOT server, and data is received either in JSON or XML depending on the accept header of the post request. By default, XML is used as reply.

Sending Data

URL: http://hwbot.org/version/api

Request parameters:

- **client**: text, required. This identifies the third party application. E.g. “Unigine_Heaven”.
- **always200OK**: boolean, optional, default false. When “true”, the response will always be 200 OK, regardless whether the API call was a success or not. This is required for some frameworks.
- **mode**: text, optional. When different use cases are possible after submission of the data file, the user will be presented with a choice what he wants to do on a landing page. This can be skipped by specifying the mode parameter to directly choose a specific use case. E.g. “analyze”

Response

The response can either be formatted in **JSON or in plain XML**. Either way they contain the **response** object contain the same fields:

- **version** = current version registered on HWBOT. Not using this exact version of the third party API might cause the submission to be rejected.
- **url** = optional, a link where the newest version can be downloaded.
- **error** = only in case of error, when there is no benchmark registered with the requested name

COLARDYN IT GCV

The response HTTP codes are:

- **200 OK**: success
- 4**00 BAD REQUEST**: required parameter missing or unknown benchmark client
- **500 INTERNAL SERVER ERROR**: uncaught error

Example response:

```
<response>

<version>2.0.11</version>

<url>http://downloads.hwbot.org/heaven/heaven-installer-2.0.11.zip</url>

</response>
```